VZDU # 31 JULY—AUGUST 1991

```
##      ##################  #####  #####
 ##    ## #       #       #  #     #     #
  ## ## ####   ####    #  ####   #      #
   ### #      #     #  #      # #     #
    #  ##### #####  ###########  ####
```

DOWN UNDER CLUB

Editor
Harry Huggins
12 Thomas St
MITCHAM 3132
03-873 1408

Treasurer
Ron Allen
2 Orlando St
HAMPTON 3188
03-598 4534



YES! FIVE YEARS OLD AND STRONGER THAN EVER.....

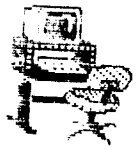
VZ COMPUTER USERS' CLUB.

BIMONTHLY NEWSLETTERS.

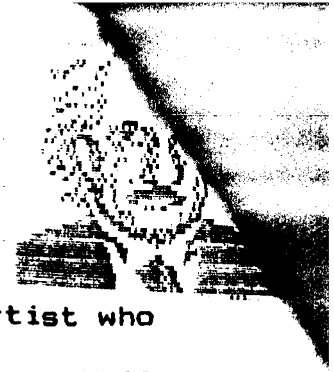
ACCESS TO VZ LIBRARY OF GAMES, PROGRAMS, HANDBOOKS.

TECHNICAL ASSISTANCE GIVEN, PROBLEMS SOLVED.

"TRADING POST" TO BUY AND SELL SOFTWARE AND HARDWARE.



FROM THE EDITOR



HAPPY FIFTH BIRTHDAY

I have been relegated to the second page, by the artist who earned the front page.

The author is Mitch Pendlebury, not yet in his teens. Well done Mitch, and thankyou. We hope to see you back in the High Scores soon. At present we are watching the duel between Peter Watson and the McLean gang. They have the limelight, but don't let them get away with it.

I have a suggestion from Peter. How about running a competition for the best design done on the computer. It's your club, so let me hear from you. How is it to be done? Anyhow, or using the utilities such as Sprite, Art Gallery, Magic Paint Graf Star etc., and in Basic or machine code. And do we Bar the "Experts"?

If it is decided to use the utilities, does eveyone have them. Some I can add to the library, but I can't add Art Gallery.

Well, don't just sit there. Grab a pen and start writing!

I may have misled many when I've mentioned memory size. I have been using a memory expansion marked 16K, in the beleif that if I used that I could make sure I did'nt print anything that would'nt run on standard memory. I was pulled into line when I gave a program to a member, and showed how it ran. He got home and phoned me to say it would'nt even load. After checking lots of things I compared his Top of Mem with mine, and suddenly realised that my 16K was indeed a 64K; mis-marked.

There was a batch came out like that, I knew, but never thought I'd be lucky enough to get one. Maybe you have one and don't know it, so check. To do so type this in without line numbers, and if you have disk drive, then take the controller out and plug the memory straight into the computer.

```
PRINT PEEK(30897);PEEK(30898) and
press RETURN.
```

You will get 2 numbers. A 64K unit will be 255 255. Any other numbers and it is as marked. A 64K will have a top of memory of 65536 bytes. Mybe you are lucky, as there were a lot that both Laserlink and Dick Smith brought in. Not just an odd one. Even if you don't use the bank switching, they still give about an extra 2K of useable memory.

As a birthday issue, you will see we have a few extra pages in this issue.

Tapes 8-9-10 are now available, and their index is elsewhere. Also a list of the books we have available for loan. If you get these, please return them promptly, as others may need them. Unless you are advanced in M/L, don't rush for Zac's programming the Z80. It has everything; if you can understand it. It is not a VZ book, but deals with the MPU that the VZ uses.

HI-RES GRAPHICS GEOMETRIC PLOTTING.

By Bob Kitch

The following program is a simple line plotting routine using the hi-res graphics screen. It was written to try and demonstrate how programming skills can be improved by following a few simple guidelines. It is a plea for more readable Basic programs.

Unfortunately published programs, in magazines, are generally poor examples of how to develope good programming style. A number of us may have taken the trouble to enter a listing from a magazine - but upon running the code have found that all is not well! A long, tedious and frustrating session of understanding the poorly constructed code follows. Often this requires that the twists and turns of the "logical spaghetti" be unravelled before debugging can commence. A usual remedy is to re-write the program from scratch.

The program LINPLOT is written using the following guidelines -

1. Clearly coded and set out - an enormous help to UNDERSTANDING.

2. The program is STRUCTURED - a good algorithm is selected and the program "flows" through initialization, to input, procedure and output sections.

3. Loops are indented for ease of identification and nesting.

4. Naming of variables is meaningful to assist maintenance and debugging.

5. Integer storage is used where appropriate.

6. No abbreviated forms of Basic statements are used.

7. Remarks are liberally sprinkled throughout to aid clarity.

8. Error capture and range checking on all input variables prevents the program from crashing.

Clear readable code is more important than the execution speed or storage requirements of a program - interpreted Basic runs like a tired snail in any case!

These guidelines should lead to code that is easier to read, understand and debug. This leads to easier maintenance, updating or expansion of your routines as your programming skills develop.

```
 10 '           ************************************
 20 '           *** PLOT A SET OF UP TO 20 LINES ***
 30 '           *** USING THE HI-RES SCREEN.     ***
 40 '           *** R.B.KITCH  22/10/85          ***
 50 '           ************************************
 90 '
100 '*** DIMENSION STORAGE VECTORS X% & Y%.
110 DIM X%(20),Y%(20):CLS:'***VECTORS TO HOLD END CO-ORDS.
115 '
120 '*** ACCEPT INPUT AND CHECK.
```

```
130 INPUT"HOW MANY LINES - MAX 20 ";LN%
140 IF LN%<1 OR LN%>20 THEN GOTO 130
150 FOR I%=0 TO LN%        :'***LOOP FOR LN%+1 X-Y POINTS.
160     INPUT"ENTER X-VAL 0-127 ";X%(I%)
170     IF X%(I%)<0 OR X%(I%)>127 THEN GOTO 160:'***CHECK ON SCRN
180     INPUT"ENTER Y-VAL 0- 63 ";Y%(I%)
190     IF Y%(I%)<0 OR Y%(I%)> 63 THEN GOTO 180:'***CHECK ON SCRN
200 NEXT I%
290 '
300 '***SET UP SCREEN AND MAIN PLOTTING LOOP.
310 MODE(1)                :'***SWITCH SCREEN TO HI-RES.
320 FOR I%=0 TO LN%-1      :'***ASSIGN MAIN LOOP FOR LN% LINES.
330     X1%=X%(I%):X2%=X%(I%+1):'***ASSIGN END POINTS TO TEMP VAR
340     Y1%=Y%(I%):Y2%=Y%(I%+1):'***ASSIGN END POINTS TO TEMP VAR
350 '***ARE THE POINTS THE SAME?
360     IF X1%<>X2% OR Y1%<>Y2% THEN GOTO 410
370     SET(X1%,Y1%)       :'***END POINTS ARE THE SAME SO PLOT.
380     GOTO 710
390 '
400 '***CALCULATE X AND Y DIFFERENCE.
410     DX%=X2%-X1%:DY%=Y2%-Y1%:'***CHANGE IN X & Y DIRECTIONS.
420 '***SEE WHICH IS LARGER.
430     IF ABS(DX%)>ABS(DY%) THEN GOTO 610
490 '
500 '***INCREMENT IY OR ALONG Y-AXIS.
510     YS%=SGN(DY%):DG=DX%/DY%:'***SIGN OF STEP AND GRADIENT.
520     XO=X1%+0.5         :'***X-AXIS OFFSET.
530     FOR IY%=Y1% TO Y2% STEP YS%:'***INITIALIZE LOOP.
540         TP=(IY%-Y1%)*DG+XO:'***TEMP REAL X-VALUE.
550         IX%=INT(TP)    :'***INTEGER X-VALUE.
560         SET(IX%,IY%)
570     NEXT IY%
580     GOTO 710           :'***PICK UP ANOTHER LINE.
600 '***INCREMENT IX OR ALONG X-AXIS.
610     XS%=SGN(DX%):DG=DY%/DX%:'***SIGN OF STEP AND GRADIENT.
620     YO=Y1%+0.5         :'***Y-AXIS OFFSET.
630     FOR IX%=X1% TO X2% STEP XS%:'***INITIALIZE LOOP.
640         TP=(IX%-X1%)*DG+YO:'***TEMP REAL Y-VALUE.
650         IY%=INT(TP)    :'***INTEGER Y-VALUE.
660         SET(IX%,IY%)
670     NEXT IX%
690 '
700 '***END LOOP FOR LINE.
710 NEXT I%:SOUND 28,6     :'***END LOOP.
720 AN$=""
730 AN$=INKEY$:AN$=INKEY$  :'***PAUSE FOR ANY KEYSTROKE.
740 IF AN$="" THEN GOTO 730
790 '
800 '***GO AGAIN?
810 CLS:PRINT"  (E) TO EXIT":'***SCREEN MESSAGE OR MENU.
820 PRINT"  (P) TO PLOT AGAIN"
830 PRINT"  (N) FOR NEW POINTS":PRINT
840 INPUT AN$              :'***ACCEPT RESPONSE.
850 AN$=LEFT$(AN$,1)       :'***CLEAN IT UP.
860 IF AN$="E" THEN STOP   :'***LOGICAL END TO PROGRAM.
870 IF AN$="P" THEN GOTO 310:'***GO BACK AND PLOT AGAIN.
880 IF AN$="N" THEN GOTO 130:'***GO BACK FOR MORE INPUT.
890 GOTO 810               :'***WRONG RESPONSE.

900 END
```

MORE I/O ERRORS.

Never a dull moment here. A phone call from a member who had an important WORDPROCESSOR file on disk, and got an I/O ERROR when trying to load. Thanks to another member who lived nearby (country miles), it was saved to another disk. His first attempt was futile, but he switched to the EX DOS in the EPROM, and it came up O.K. I would like to know why that was so, or was it just co-incidence?.

During writing, the item was saved as ITEM-1, ITEM-2 etc. For the last saving the disk was full, so the previous items were ERAed.

I do not advocate using that method. The ERA does not wipe the disk clean. It merely removes the T-B-W-D or whatever, and adjusts the track map. The rest of the text is left there to be overwritten. The UPDATE command is merely an ERA"...":SAVE"..." command and the same state exists.

I will quote from the DOS handbook:-

'Diskettes that have had a lot of creation and deletion activity become fragmented, because space is not allocated sequentially.-------.A fragmented disk can cause degraded performance due to excessive head movement and rotational delays involved in finding, reading, or writing a file. Unquote.

I know the disks get blamed. I have a collection of over 500 disks, and have only struck 2 BAD disks. Both were TOP BRAND names.

The foregoing does not apply to a disk that has been re-formatted. That IS wiped clean.

Some causes of I/O errors are:-
Leaving the disk in the drive with the door closed during reset, Power up or powere down, either intended or when the computer does it without command. After saving or loading, open the door. Make a habit of it. *REPEAT* Make a habit of it.
Bending of the disk. Be careful when putting it in the drive.
A power surge during saving or loading
Excessive heat
Dust or fingers on the disk surface.
Excessive pressure in storage. An extra disk where there's no space for it.
High humitity and anything else you can think of.

The way around?

Duplicate copies on separate disks.

Fuzzy Logic

by Jason Oakley

The term "Fuzzy Logic" was coined by Professor Lofti Zadeh of the University of California. It was used at Berkley in a seminal paper in an academic journal, Information and Control, in 1965. The concepts were the work of a Polish mathematician, Jan Lukasiewicz, in the 1920s and therefore, although only widely heard of in the last few years, has been around for quite a while.

Professor Zadeh wanted to use concepts which cannot be precisely defined in mathematical terms. Examples of these concepts include: tall, fat, beautiful, large and even middle-aged. Most of the terms are abstract and depend on who is saying them. for instance, a person might look beautiful to someone - but not to someone else; a four-foot high tree may seem small - for an oak - but if it was a bonsai tree it would be quite large. The terms change according to the individual.

Aristotle's logic Law of the Excluded Middle becomes redundant and fuzzy theory lets objects be members of contradicory groups. There only exists a greater probability that one thing belongs to a set than to another. eg. a day may have a 70 per cent chance of being too hot when the temperature is 35 degrees celcius if you wish to play sport, but it would be just the right temperature for going for a swim.

Fuzzy logic allows computers to make decisions, if programmed to a persons' preferences. It is more efficient (Toshiba's voice-controlled lift system uses fuzzy logic to work out how long people have been waiting, and favours those who have waited for a minute or more). It is cheaper (Mitsubishi's Beaver Warp Inverter Air Conditioner uses 50 "fuzzy rules" to keep a constant air temperature and uses 24 per cent less energy than traditional air conditioners).

As a result of this, the Ministry of International Trade and Industry (MITI) set up the Laboratory for International Fuzzy Engineering (LIFE) with $us70 million for five year's research funding.

From the way fuzzy logic is going, it shouldn't be too long before we will have the machines around us running on fuzzy logic - Televisions, Dishwashers, Toasters, Video cameras. How about a radio which "knows" your favourite music types and searches the local radio stations for your favourite piece of Beethoven, Crowded House or just any Reggae tune. Sounds too good to be true? Not at all! already tests are being done to bring about this sort of thing.

Although these things will come out, there will - of course - be a market for machines which will still be operated manually. There will always be people who like things done the good Old Fashion way.

fuzzy logic source - jack schofield - smh 20 may 1991.

ADVENTURE GAME WRITING - THE VERB SUBROUTINES

The next section in the program branches the program off to the various verb subroutines. This is normally done by an ON VB GOSUB 800,800,800,800,930,980,1060... statement. Unfortunately the standard VZ does not support this command unless one of the various Extended Basics available is used. The most common way around this is to use a great pile of IF... THEN statements.
eg.

```
IfVB>0ANDVB<5THENGOSUB800
IFVB=5THENGOSUB930
IFVB=6THENGOSUB980
IFVB=7THENGOSUB1060
etc.
```

There is a way to simulate the ON... GOSUB command from standard VZ Basic, but once this has been done, these particular lines can't be edited. For this reason it is important that the line number at the start of each verb subroutine is known before you do this. Therefore all of the verb routines must be designed beforehand.

(There is no need to type the program in in order of line numbers. Just type in each particular section as it interests you - I generally start with the data statements even though these are at the end of the program.)

The verb subroutines are not difficult to write so long as you make a few prepartions beforehand. You should have your adventure map in front of you, and this should show the locations of the gettable and fixed objects. You should list your verbs and all of your nouns, numbered in the order that they appear in X$ or Y$. (see the map in VZDU Nov/Dec 1990.) You should also have a list for your F (flag) array, showing the use of each element. There should be one place for each gettable object, and this should indicate whether it is visible. As you come across a use for an element, write it down on your list so you have it for future reference - you will almost certainly need to use the same flag elsewhere in the program. For example, if F(19) is set aside for a locked door, not only is this altered by the UNLOCK routine, but when you are moving around, the move routine has to check whether you can walk through the open door, or are trying to pass through the locked door by supernatural means.

I usually write all of the verb routines out onto paper first before I type them into the computer. This enables you spot possible problems and errors before you type them into the computer. If you find a bug while running the program, especially one which does not cause an error message, it is easy to simply look through your listing to find the exact location. It helps you to also add in a few extras, such as sound routines as required. If you decide to do this, make sure you leave a gap of at least 10 between each line number. You may find that one line may turn out to be two or even three or four when you type it in, due to the VZ's 64 character restriction. You may need to add in lines that you forgot to include earlier, or you may want to add some sounds or other things when the main body of the program is finished.

However once you have everything prepared the verb routine lines can simply be "read" in the same sort of manner as sentences. Here are a few examples from the EXAMINE/LOOK routine. (Note the use of H as R*100+B from the last edition.)

```
1120 R$="YOU SEE NOTHING SPECIAL ABOUT THE "+T$
```

(This line is included at the start. If there is nothing special
the object the player has asked to examine, this message i
altered.)

```
1190 IFH=819THENR$="IT SCRATCHES YOU":S=S-3
```

(If you are in room 8 - the room containing the cat - and wish
examine this cat, reply "It scratches you" and remove three streng
points.)

```
1230 IFH=222ANDF(24)=1,R$="THIS IS MUCH MORE INTERSETING":F(4)=0
1240 IFH=222ANDF(24)=0,R$="AARCH! ITS YOUR ENGLISH NOVEL":F(24)=1
```

(If you are in the lounge and wish to examine the bookshelf, and have
not examined it before, reply is the message about the english novel,
and update F(24) so that you have looked at the shelf before. If you
have examined the shelf before, reply "This is much more interesting",
and make the computer book visible.)

```
1280 IFH=2325THENF(11)=0: R$="SOMETHING HERE, ROVER!"
```

(If you are by the bushes, and examine them, the reply is "Something
here, Rover!" and make the bone visible.)

```
1320 RETURN
```

(Don't forget this must be placed at the end of each verb routine.)
This is a typical routine - LIGHT

```
1560 R$="FOOLING AROUND WITH THE "+T$+"ISN'T GOING TO HELP"
1570 IFB<>14THENRETURN
1580 IFF(27)=1THENR$="IT'S ALREADY ALIGHT, YOU MORON":RETURN
1590 IFC(13)<>0THENR$="YOU DON'T HAVE ANY BATTERIES":RETURN
1600 R$="IT CASTS A BRILLIANT BEAM":F(27)=1:IFF(21)=1,F(21)=0
1610 RETURN
```

If the player tries to light anything but the torch, this input is
obviously ridiculed. The input is also rejected if the player tries to
light the torch when it is already lit, or when there are no
batteries. (The situation where the player has no torch is dealt with
in the error checking routine discussed last issue.) If the player
wants to light the torch, has the batteries, and it is not yet lit,
F(27) is altered to show it is lit, R$ is changed to "It casts a
brilliant beam" and if the player is in a room where nothing can be
seen (F(21)=1), the room is lit up (F(21)=0).

In some lines, the list of conditions for IF... THEN statements is
so long that there is hardly any room for the actions which are
carried out if the condition is true, due to the 64 character line
restriction. Rather than type the condition on the next line again, in
the first line you set a variable (I used the letter U) to equal one,
then on the next line type IFU=1THEN..... Don't forget to reset U to
equal zero again afterwards.

eg.

```
1980 IF(F(22)=R AND B=18) OR H=819THENU=1:S=INT(s/2): R$="THE "+T$
1990 IFU=1THENR$=R$+" DOES NOT TAKE KINDLY TO BEING EATEN, "
1995 IFU=1THENR$=R$+"AND SO BEGINS TO EAT BACK":U=0
```

(These lines are part of the EAT routine. Note that the dog in this
program can move around at random, patrolling the area in rooms four,
five and nine. It will also follow you if you make friends with it
somehow. It's room location is stored in F(22). If you are in the same
room as the dog and try to eat it (a very silly action!) or are in the
cat's room and try to eat the unfortunate tom (equally silly!), your
strength will be halved due to the retaliation, and the message from
the above lines is put into R$. this was one of the irrelevant or
silly replies to irrelevant or silly combinations of words that I

mentioned in the first article in this series.)

With the use of the U variable in the previous lines, it is a good idea to use the same variable for every time you need to use one for this purpose. Space is reserved in the memory for every different variable you use, so it is best to use as few as possible.

With your lists of verbs, nouns and flags, your map, and the listings of other adventure programs, you should be able to design most of your verb routines yourself. However there are a few other verb routines that should be examined in some detail.

MOVING AROUND

```
800 D=VB:O=R:IFD>4THEND=0
```

(Lines 810-860 are left for things that prevent the player from moving in the direction they want to, in spite of there being an exit there. This may be a locked door or a magical force field. If you have different levels in the game, these lines also deal with moving up or down. This game doesn't have any up or down, but what you would do is set aside two extra verbs - U and D. You would then use lines like the following:

```
820 IFR=29ANDD=6THENR=59:R$="OK": RETURN
830 IFR=59ANDD=5THENR=29:R$="OK": RETURN
```

Most other lines would resemble

```
810 IFD=1ANDR=21ANDF(19)=0THEN R$="YOU CAN'T. THE DOOR IS LOCKED"
```

But your lines would be slightly different for different situations.)

```
870 IFMID$(D$,D,1)="0",R=R+VAL( MID$("-505-101",D*2-1,2): R$="OK"
880 IFO=RTHENR$="YOU CAN'T GO TO THE "+Z$(D+9)
```

Line 870 is quite complex so we will use an example to explain it. Assume the player entered "N". In this case D would equal one. If the first character of the movement code is a zero then there is an exit to the north. Therefore the value of the first two characters in the "-505-101" string is added to R, so in effect 5 is subtracted. If the move was south 5 would be added; if it was west, one subtracted, and if it was east, one would be added. The message "OK" is put in R$, to change it from "I need two words".

If the movement code revealed that there was no movement possible in the desired direction, then O would remain equal to R so the message in R$ would be changed to "You can't go to the north" (or whatever direction).

```
890 IFF(27)=0AND(R=15ORR=20)THEN F(21)=1ELSEF(21)=0
```

(If your torch is not lit, and you are in room 15 or room 20, then make the room dark. If not the room is light.) If you wish to make it so that the player can't enter a dark room without a light at all, this is the sort of line you would use.

```
890 IFF(27)=0AND(R=15ORR=20)THEN R=O:R$="YOU NEED A LIGHT TO GO THERE"
```

The rest of the lines in the move routine after this deal with things that might happen to the player as they enter a new room, such as falling down a hole, being attacked by a hideous monster, or being seriously annoyed by a bunch of crickets chirping at him.

INVENTORY ROUTINE

```
980 CLS:PRINT"YOU ARE CARRYING ";:U=0:T=0:R$="OK"
990 FORI=1TOG:IFC(I)=0THENU=U+1: NEXTELSENEXT
```

```
995 IFU=0THENPRINT"NOTHING.": GOTO970
```

This part of the routine checks to see how many items the player is carrying. If none are being carried the player is notified that he is carrying nothing.

```
1000 FORI=1TOG:IFC(I)<>0THENNEXT: PRINT". ":GOTO1040
1010 L=I*5+4695:GOSUB5300:READO$: PRINTZ$(ASC(O$)-44);
1015 PRINTRIGHT$(O$,LEN(O$)-1);: T=T+1:IFT<U-1THENPRINT", ";
1020 IFT=U-1THENPRINT" AND ";
1030 NEXT:PRINT"."
1040IFC(5)=0THENPRINT"YOU HAVE";F(25);"ITEMS OF FOOD LEFT."
1050 U=0:GOTO970
```

In this part if the player has a certain object, the line number containing its description is calculated. The description is then read from the data. It is printed by decoding the number in front of it. (see the section on displaying the environment.) The rest of the description is printed followed by a comma, an " and " or a full stop - whichever is appropriate. If the player is carring the food, (s)he is told how many items are left. (When picked up there are ten items, and this is decreased each time the player EATs it.) Finally to save memory, the routine goes to the end of the HELP subroutine at 970 - this part is required for both routines, but is self-explanatory.

```
970 PRINT@480," press any key to continue.....";:'INVERSE
975FORI=1TO10:A$=INKEY$:NEXT: IFA$=""THEN975ELSERETURN
```

GET ROUTINE

```
1060 IFB=0ORB>gTHENR$="YOU VANDAL! YOU CAN'T TAKE THAT":RETURN
1070 IFC(B)<>RORF(B)<>0,R$="I CAN'T SEE THE "+T$:RETURN
1080 IFW+W(B)>S/2THENR$="YOU CAN'T CARRY JUNK THAT HEAVY":RETURN
1090 C(B)=0:W=W+W(B):R$="OK. YOU TOOK THE "+T$
1110 RETURN
```

(Note that lines 1093,1095 and 1100 are only relevant to the demo adventure.)

The player is informed and the input rejected when an attempt is made to take a non-gettable object, a gettable object that isn't there, has already been taken, or is not visible. If the combined weights of all the objects the player is carrying, plus the object being picked up exceeds half of the player's current strength, the object is deemed too heavy for the player to carry, and can't be picked up unless something else is dropped or strength is somehow increased.

If none of the above happens, the player can pick up the object and the weight being carried is adjusted accordingly.

LEAVE/DROP ROUTINE

These can be both covered by the one routine despite the fact that things that are DROPped are more likely to tend to splatt, etc.

```
1330 IFB=0ORB>GTHENT$="YOU AREN'T CARRYING THE "+T$:RETURN
1340 C(B)=R:R$="OK":IF(B=13ORB=14) ANDF(27)=1THENF(27)=0
```

```
1345 IFB=10THENF(23)=0ELSEIFB=7 THENF(31)=0
1347 W=W-W(B)
1350 IFVB=11THENRETURN
1360 IFH=2415THENR$="VANDAL! YOU BROKE THE GARDEN GNOME":RETURN
1370 IFB>14ANDB<18THENS=S-W(B)/5: R$="OUCH! IT FELL ON YOUR FOOT"
1380 RETURN
```

If you leave the torch or batteries behind, and the torch is lit, it is then extinguished. If you drop the headphones or gasmask while you are wearing them, it is assumed you removed them first. The appropriate weight must be subtracted from the total weight carried.

At line 1350 the LEAVE subroutine RETURNs so anything happening afterwards applies only to the DROP routine. In this program if you drop the rock near the garden gnome you will smash it, otherwise if you drop the rock, the fridge or the washing machine (yes, the last two can actually be picked up if you are strong enough) they will fall on your foot, and, quite natually, hurt it. Both of these will not happen if you LEAVE these objects.

Note for all verbs in general, if an object disappears off the face of the earth, by being destroyed in some manner for example, you let it's C flag equal the number of rooms plus one. When the food is all gone in this adventure C(5)=26.

DIRECTING THE PROGRAM FLOW TO THE RIGHT VERB

This section shows how to use the ON GOSUB command anywhere in your program without an extended BASIC. Make sure you have your program saved on tape or disk, as you could destroy it if you make a mistake.

Firstly type the following somewhere where it will not be in the way of any of the other program code. eg line 10000

```
10000 DATA237,91,33,121,205,44,27, 210,217,30,237,67,33,121,201
10005 L=10000:GOSUB5300
10010 FORI=31290TO31304:READA: POKEI,A:NEXT
10020 POKE30862,58:POKE30863,122
```

Type GOTO10000 to run this routine.

Type your ON GOSUB lines exactly as they would appear normally except instead of the word ON, use some sort of dummy symbol. I usually use a hash symbol (#).

```
420 #VB GOSUB800,800,800,800,930,980, 1060,1120,1120,1330,1330
```

(list the line number at the start of the first verb first, the second verb second, etc.)

You won't be able to fit all of the verbs line numbers on a single line because of the 64 character line restriction. To get around this we would start another line with #VB-11 GOSUB... This creates another problem though. When VB is 11 or less, it will GOSUB the appropriate routine, but when it RETURNs, it will come across an ONVB-11 GOSUB... statement. Needless to say, ON GOSUB doesn't like negative numbers very much, so they cause ?FUNCTION CODE ERRORs. Therefore in between each # GOSUB, we must put lines such as:

```
425 IFVB<12THEN450
430 #VB-11 GOSUB1390,1450,1480,1560,1620, 1650,1680,1720,1790
```

```
435 IFVB<21THEN450
440 #VB-20 GOSUB1820,1860,1970,2050,2250, 7000,7100
```

Make sure you have every line perfect for your program, because once the lines have been turned into proper ON GOSUB statements, you will not be able to LIST or edit them properly.

If you have not loaded the find line number into memory by using the subroutine at 10000, do this now. If you have run the program since doing this, you must change the values POKEd into 30862 and 30863 back to the values in line 10020 - ie 58 and 122 respectively.

For every line of ON GOSUB, type PRINTUSR(line number)+4

eg PRINTUSR(420)+4

This should return a value somewhere above 31469. If it is greater than 32767 - in most cases it will be, you must subtract 65536 from it to obtain your value. This value is the location of the first byte of code in your line. (You should write this value down in case you need to edit the line later on.) From here it is just a matter of POKEing the token for ON into this position - the value of this token is 161.

eg

```
PRINTUSR(420)+4
35000
```

When 65536 is subtracted from this the result is -30536. (Note that the 35000 is not the actual value - it is just an example to illustrate how this is done.)

```
POKE -30536,161
```

Now if you list line 420 you will see the line number but you will not see any of the code - just a blank line.

Once you have done this for all your ON GOSUB lines, everything should work properly. The lines from 10000 onwards are now fairly useless so you can delete them. If you find you have put the wrong line number in one of the statements, or made some other error, POKE 35 into your location value for this line. Now the line will not work but you will be able to LIST and edit it. Once you have fixed it up, POKE 161 into the appropriate location once again.

There is another section between this and the verbs which deals with things that happen irrespective of what verb the player entered. This sort of thing deals with decreasing the player's strength due to tiredness either by a constant amount, or in this case depending on how much is being carried. Normally the battery life (or life of whatever light source you use) would also decrease.

```
eg IFF(27)=1THENTL=TL+1: IFTL=50THENR$=R$+". YOUR BATTERIES ARE DYING"
IFTL=60THENF(27)=0:TL=61:R$=R$ +". YOUR BATTERIES ARE DEAD"
```

Other things which happen include the dog stealing your food, the dog chasing the cat if both are in the same room, wear and tear on you caused by being in the same room as your brother, the dog moving around at random, etc. It also checks to see if your strength has become zero. If it has, you have either died or become too weak to continue. If not it returns to the main part of the program.

By now, if you have a copy of the demo listing and have written a plot, you should have enough information to put your own adventure together. However in the next issue we will look at a few extra routines for advanced programs.

REPLACING PRINTER RIBBONS

By JOHN LUXTON

Who has a printer that uses a half inch or 12mm plastic base carbon tape? Who would like to be able to reload the cassette and save a bundle? Well, if your printer uses a cassette which exposes about 26cm of tape which the print-head works on, or maybe other configurations, I may be able to help.

My printer is a Compute Mate 130, model CPB80. According to the box of a Pelikan ribbon I once bought, this cartridge is used by quite a number of printers including a Commodore 4023 P, a Sekonic, Shinwa and probably others. A glance at a catalogue reveals many other similar types. Also the cartridge I use has a projecting knob at the pinch roller end for manual rewinding. The opposite side has the socket which is engaged by the printer mechanism for advancing the ribbon.

last year when we were in Sydney I got a yellow pages to see if I could find a supplier of bulk ribbon to refill spent cartridges. After a few discouraging calls I discovered a firm, Lazarus Ribbons, of 70 Wolseley Rd., Mosman, 2088, phone (02) 960 2737, who would sell me a 1000 foot roll of ribbon for a reasonable price. They were awaiting stock from the U.S.A., so in due course I received the roll of ribbon posted to my home for $29.73 all up!

I first experimented by removing the top of the cartridge by the judicious use of a knife. These tops are held by dowels, or occasionally with small screws and dowels, but can be removed with care. However this method is not recommended, as springs and things can cause problems. The better way to go about the job is to pull out the ribbon from the delivery end until the cartridge is emptied.

Before proceeding a few things are required to do the job. A video/audio tape splicing jig, Tandy cat. No. 44-9570 at about $6.95 is handy. I prefer the wider 3M Magic Mending Tape, but half inch video splicing tape is O.K. A sharper razor blade than supplied by Tandy is also useful. Winding the new ribbon into the cartridge can be done manually, but takes time. I use my newly acquired Makita cordless drill come screwdriver, set in reverse. It is slow enough, but does the job very efficiently.

When the tape is all emptied from the cartridge, cut the tape across the front, and splice on the end of the replacement tape. Make a good splice and trim any excess sticky tape, then pull the new ribbon through the cartridge from the delivery end. If your splice was faulty and breaks inside, you will have to take the lid off and feed the tape through, making sure you don't end up with springs and pinch rollers everywhere. Pull out a foot or so of the new ribbon, then devise a way of measuring off the required amount from the roll. Use clean boxes or such to hold the loose ribbon and keep it clean. I have found that the cartridges I use hold from 25 to 30 metres. It seems to vary a bit. You can best work out your own method of measuring the required length.

Now to load the new ribbon into the cartridge. The winder knob on my cartridges goes anti-clockwise, so I have to set reverse on the drill, fasten the chuck to the winding knob without using undue force. Then I hold the cartridge vertically with the delivery end between my knees, and holding the drill in my right hand commence loading, guiding the tape through my left hand to avoid any kinking. When a couple of feet of ribbon are left, unchuck the drill. Then carefully splice each end, taking care not to twist the ribbon, and putting the splicing tape on the shiny side. If using the wider tape, carefully trim with sharp scissors any sticky tape either side of the ribbon, then wind the excess into the cartridge. I sometimes again chuck up the drill and run the ribbon through to check on the splice, but if quality splicing tape is used, there should be no worry. At the moment I have reloaded 4 cartridges, and find that I can do one under 20 minutes.

The savings are obvious. At least 10 cartridges can be reloaded from a 1000 foot roll of ribbon. Depending where I buy ribbon cartridges, and the brand, the cost can average,say, $20. That's an outlay of around $200 against the $30 I paid for the roll of ribbon. So far I can't vouch for the quality of the ribbon, but was assured it was best quality from the U.S.A.

I found it advisable to make up a couple of cheeks the size of the roll from masonite. The hole in the centre of the roll was one and a half inches, so I cut a circle from half inch pine board with a hole cutter, and used the piece for the centre of the roll. Fortunately I have a lathe to finish and sand the cheeks and centre to size. Possibly the ribbon can be spooled off without spilling, but I took no chances. A quarter inch machine screw and nut finished off the job.

As I indicated, I can only speak for the cartridge used in my printer, but I should imagine that most types of cartridge containing plastic half inch ribbon can be dealt with this way. Could be an opening for a cottage industry!.

How to Make You Own SPRITE Programs

By Peter Ross

Have you ever wanted to make your own graphical game like Dawn Patrol.?
When you use Basic,it is too slow to move the charactems on the scmeen.
If you use Trent Georges*SPRITE EDITOR*,you can make a simple game in a few hours.
First load the *Sprite Editor*. This tape has two different screens.
Screen 1 is the low resolution,it has a screen where you draw your sprite.

Screen 2 is high resolution,it gives you a demo of your sprite of how it will look.
Use the controls next to the view screen to draw the sprite.

After you have assembled it PRESS X for a demo,PRESS X again to go back to low resolution screen.
When you are fully satisfied with your sprite,Press SHIFT and 8 ,this will wipe the sprite editor,so you can start typing your game.

The first line of any Sprite Program must be;

```
10 POKE 30362,0 : POKE 30363,124
```

Now you can type in your background,it can be mazes,grids,etc.
It is a good idea to draw your background first on some graph paper.
Once you have typed in the background and want the sprite to appear again, type in ;

```
30 POKE 31631,2 : D=USR(256*X+Y) : POKE 31631,0
```

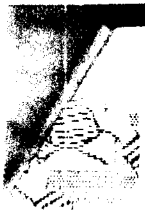
X and Y are the positions where you want the sprite to appear.
Now type in;

```
40 D=USR(-1)
```

This line will update the position of the sprite,if the keyboard or joystick has been operated.
Now consult the "Sprite Editor Book" that comes with the tape for;

Changing sprites
Collision testing
Aim & Fire testing





LETTER TO THE EDITOR



Dear Editor,

Despite the fact that DSE stores no longer support VZ hardware and peripherals, and their software is fast drying up, a number of very active Users Groups still exist. Furthermore, a number of hard-working and dedicated people have/are developing some excellent software for the VZ.

An excellent range of Utilities now exist for the VZ and I thought it might be useful to identify the types of software available to Users and to mention the sources of these items. The opinions expressed are of course mine, but they are based upon the actual use of the various packages mentioned.

Most of the "new" and high quality software is disk-based - sorry for the tape-based Users but a comparable range does exist for these systems. All of the software is supplied with good manuals.

The Best Single Piece of software available for the VZ, and probably the Most Useful, is the QUICKWRITE II Text Editor developed by Milburn and sold by VSOFTWAREZ. The Word Processor is superb and the manual excellent. Well worth $40. 64K Memory Expansion essential.

The Best Database program is Laserlink's MARK 64 DATA BASE available from me for $30. A 32K version is also available as the one mentioned, needs 64K.

The Best Spreadsheet is LASERCALC by Kitch and available from me for $30 - manual included.

The Best Graphics package is ART GALLERY by Bruce Kitch and available from me for $20 incuding manual. (A multi-disk version of PRINT SHOP will soon be available.)

That covers the four most useful categories of software, but an important set of Utilities exist for the Low-level Programmers.

The best Assembler is Laserlink's DISKED EDITOR ASSEMBLER by Harwood and avaialable from me for $25.

The best Disassembler is the DISASSEMBLER DISK by Peter Hickman and available from him for $25.

The best Monitor-Debugger is the DISK DOCTOR TOOL KIT by Kitch and is available for $30. A disk sector reader is included.

For copying programs Larry Taylor's COPYPRO4 is second to none and available from VSOFTWAREZ for $20.

As most people use Epson compatable printers, Larry's PRINTER PATCH V1.4 is invaluable at $10 from VSOFTWAREZ.

The best Extended BASIC available is undoubtedly Laserlink's XB by Russell Harrison and available from me for $30.

The only BASIC Compiler available is Laserlink's VZBASCOM by Kitch and is available from me for $30.

The Best Other Language available on the VZ is the Public Domain CHIP-8 Interpreter.

Well there it is! A very useful set of software for $300 that turns the VZ into a very smart Z80-based machine.

Regards to all

Bob Kitch

Bob has given a good account of programs available for disk users, but "nary a word" for the tape users. As most of our members use tape, I'll fill in and add my opinion to Bob's, some of which differ.

The greatest boon to the VZ users must undoubtedly be Russell Harrison's X B., closely followed by Larry taylor's Printer Patch; though not necessarily V1.4. as it will not reside with X B. But PP V 1.2 will.

These P.Patches don't suit all printers though. GP100 does'nt need one. Some Tandy and OKI printers need a different patch, which are available from Larry Taylor.

Unfortunately X B is for disk use only, but there are others available for tape users; one by Onley and one by Obrist. They can be used with either tape or disk.

The Quickwrite series are only for Disk, and generally require 64K memory. However there is a good Wordprocessor available for tape, and with the addition of a patch, are very suitable for disk. I use one exclusively. It only needs 16K memory expansion. There is also a Wordprocessor cartridge for the VZ, that will run as well on the 200 as on the 300, and as it has it's own memory does'nt need an expansion.

The same Editor/Ass. Bob refers to is available for tape. Peter Hickman's dissassembler is only useable on disk, but there is on that works with tape or disk.

The Basic Compiler is the only one, and can be used with tape or disk, But it has to be loaded from disk. It is limited to about 10K programs because of lack of memory.

The purpose of a compiler is to convert a HIGH LEVEL language (Basic) into machine language, and save it as such. This means that it does not have to be INTERPRETED each time it is run, which means it runs faster.

In general though, basic with machine code subroutines for such things as sorting, runs quite fast enough.

For disk users there is an Extended Dos program with some very useful commands in it. There is also a DOS EPROM that goes in the disk controller.

The BASIC EPROM replaces the VZ ROM and gives the range of commands that the TRS 80 had. Both those Eproms are available from Bob. [ED.]

OUR LIBRARY BOOK LIST

TITLE	AUTHOR	POSTAGE	DEPOSIT	LOANTIME	REMARKS
55 ADVANCED BASIC PROGRAMS	WAATSON	$2.80	$20.00	1 MONTH	
57 PRACTICAL COMP. PROGRAMS	TRACTOR	$2.80	$20.00	1 MONTH	ADVANCED MATHS.
ASSEMBLY LANGUAGE PROGRAMMING	STEVE ONLEY	$2.80	$40.00	2 MONTHS	ABOUT THE BEST AVAILABLE
BATTLE GAMES	USBORNE BOOKS	$1.80	$10.00	1 MONTH	
COMPUTER AND VIDIO GAMES	USBORNE BOOKS	$1.80	$5.00	1 MONTH	TELLS HOW TO WIN
COMPUTER CONTROLLED ROBOTS	POTTER	$1.80	$5	1 MONTH	
COMPUTER HANDBOOK	USBORNE	$2.80	$15.00	2 MONTHS	
COMPUTER SPY GAMES	USBORNE BOOKS	$1.80	$5.00	1 MONTH	
CREEPY COMPUTER GAMES	USBORNE BOOKS	$1.80	$5.00	1 MONTH	
EXPANDING YOUR MICRO	USBORNE BOOKS	$1.80	$5.00	1 MONTH	
EXPERIMENT WITH YOUR COMPUTER	USBORNE BOOKS	$1.80	$5.00	1 MONTH	
FANTASY GAMES	USBORNE BOOKS	$1.80	$5.00	1 MONTH	
FIRST BOOK OF PROGRAMS	DSE	$1.80	$15.00	1 MONTH	ELEMENTRY
FIRST BOOK OF THE COMPUTER	USBORNE	$2.80	$10.00	1 MONTH	
FURTHER PROGRAMMING	TIM HARTNELL	$1.80	$15.00	1 MONTH	
GETTING STARTED	HARTNET & PRINGLE	$1.80	$15.00	1 MONTH	
GIANT BOOK OF GAMES	HARTNET & PRINGLE	$2.80	$25.00	1 MONTH	
ISLAND OF SECRETS	TYLEE & HOWARTH	$1.80	$5.00	1 MONTH	
MACHINE CODE FOR BEGINNERS	USBORNE BOOKS	$1.80	$10.00	2 MONTHS	
MYSTERY OF SILVER MOUNTAIN	OXLADE & TATCHELL	$1.80	$5.00	1 MONTH	
OMNIBUS	HARTNETT & PRINGLE	$2.80	$25.00	1 MONTH	PLEANTY OF GAMES TO TYPE IN.
PROGRAMMING H/BOOK BASIC & ASS	USBORNE BOOKS	$2.80	$15.00	2 MONTHS	
PROGRAMMING HINTS AND TIPS	JOHN D'ALTON	$1.80	$10.00	1 MONTH	USEFUL FOR BEGINNERS
PROGRAMMING IN Z80 LANGUAGE	HUTTY	$2.80	$30.00	2 MONTHS	A GOOD BOOK TO START ASSEMBLY.
PROGRAMMING THE Z80	RODNEY ZACS	???????	$40.00	2 MONTHS	THE LOT;POST $4.30 TO $7.85
PROGRAMMING TRICKS & SKILLS	USBORNE	$1.80	$5.00	1 MONTH	
SPACE GAMMES	USBORNE BOOKS	$1.80	$5.00	1 MONTH	
THINGS TO DO WITH YOUR MICRO	USBORNE BOOKS	$1.80	$10.00	1 MONTH	
VZ 200 TECHNICAL MANUAL	DSE	$1.80	$25.00	1 MONTH	
VZ 300 TECHNICAL MANUAL	DSE	$2.80	$25.00	1 MONTH	
WEIRD COMPUTER GAMES	USBORN BOOKS	$1.80	$5.00	1 MONTH	

INDEX TO VZDU # 1 TO # 31.

News on the VZED front

Video Technology (the company which made the good ol' VZEDs) has announced that they will be releasing a computer called IQ Unlimited in the United States for $us200. It will be released in August. The IQ will run on a television screen. It will contain built-in programs such as: Spreadsheet, Database, Word-processor with Spell-checker, Calculator and a Drawing program. The IQ will run on four C-size batteries or a Wall socket Adaptor and will contain 128K of memory.

Although a computer for under $300 seems like a good idea (remember the vzeds were originally $aus199 - later $aus99?), this computer is not expected to do well as "real PC's" with monochrome monitors will be available for a little extra money around the same time.

SHUFFLING

In VZDU $ 17 I published a routine of this name sent in by Ian Niedzwierki. He set some "Homework". Until now noone has done their "Homework", but this time a junior member has come up with this. Congratulations!!. He has issued a challenge. Alter it to deal for a 4 handed game.!!!

Come on. This member is only 12. Don't let it be said.(Ed.)

```
10 CLS:CLEAR(1000):DIMAB(52),C$(52),S$(52)
20 DIMS(40),C(14),AA(4,30)
30 PRINT@75,"SHUFFLING":FORX=1TO1500:NEXT
35 PRINT@75,"          ":PRINT@75,"DEALING":PRINT:PRINT
40 FORX=1TO500:NEXT:FORA=1TO5
50 S=RND(4):C=RND(13)+1
55 IFAA(S,C)=1,50ELSEAA(S,C)=1
60 IFS=1,S$="CL"ELSEIFS=2,S$="DIA"ELSEIFS=3,S$="SP"ELSES$="HRTS"
80 IFC>10GOSUB170ELSEC$=STR$(C)
100 PRINT"       ";C$;"  ";S$:FORX=1TO500:NEXT:NEXT
110 PRINT:PRINT:INPUT"ANOTHER HAND? Y/N";V$
120 IFV$="N"THENENDELSECLS:GOTO35
170 IFC=11THENC$=" J"
180 IFC=12THENC$=" Q"
190 IFC=13THENC$=" K"
200 IFC=14THENC$=" A"
210 RETURN
```

This will deal 10 hands of 5 cards each, without shuffle. If you want to shuffle between hands the alter line 120 to:-

```
120 IFV$="N"THENENDELSECLS:RUN
```

(Ed)

THREE NEW TAPES

TAPE-8**	TAPE-9***	TAPE-10**
********	*********	*********
ATW T	BLOCKPUZ B	DEMON AD T
CASTLE G T	COMPOKER T	FBI INST T
DOG RACE T	CRYSTALS T	FBI-2001 T
GAME INS T	DIAMOND B	INTRO T
GAME' IN T	FACTORY B	LABYRNTH T
KENO-A [illegible]	FROG T	LAND SEC T
MARBLES T	GOLF.. T	MERK INS T
MOON AGE T	INST E.T T	MERK LDG T
POKER T	KILLER T B	POKIES T
RACE SEL T	LEAPFROG T	ROULETTE T
ROADRACE T	MASTER T	SNAKEY-1 T
ROULETTE T	POKER T	SNAKEY-2 T
S.MOUNTN T	POKIES T	TOP 20 T
SOLO T	RUBICUBE T	
SPYCATCH T	SHUFFLIN T	
TARGETS T	TOTE RAC B	
YAHTZEE T	TRAP NUM T	
	U/O 1 T	

THE DIRECTORY LABEL UPDATE

WRITTEN BY D.MITCHELL

NOW we can formatt single tracks and can formatt the directory track ,we need a way to restore the directory information.

This program is an adaptation of the ' LABEL ' program i wrote earlier. To use this program you will need to know the FILENAME the START TRACK and SECTOR and the START and END ADDRESSES , these are entered in HEX.

Once this information is entered it is written to disk and if the track address is above 1 the program will update the track map showing that those sectors used by the program are not available. This program can still be used to LABEL your directories by placing zero's or by pressing the return key for the TRACK , SECTOR , START and END ADDRESSES.

Labels may be up to 10 characters long and you can place up to 120 labels into the directory if you wish. By typing DIR....... as a label the directory will be printed to the screen.

Move the cursor past the quotes displayed on the screen , it is so the delimitor used to separate the filetype and filename.

Those of you that have the previous ' LABEL ' program will need only to modify that program.

The others will have to enter the two programs listed below , the first program places the machine code into line ten , this is for the disk operations.

When the first program is run line ten is the only line left , then all that is required is to type the second program into memory from line twenty.

PROGRAM ONE THE MACHINE CODE

```
 10 REM...................................................................
 11 REM...................................................................
 12 REM...................................................................
 20 FORI=31470TO31632:READA:POKEI,A:B=B+A:NEXT
 30 IFB<>17462THENPRINT"ERROR IN DATA":END
 40 POKE31465,143:LIST
100 DATA229,243,205,8,64,14,50,6,255,4,205,56,64,219,19,183,62
110 DATA4,250,14,64,58,33,121,183,32,18,205,47,64,183,194,14
120 DATA64,34,25,123,205,11,64,251,225,201,201,254,205,50,64
130 DATA183,194,14,64,205,17,64,183,194,14,64,58,41,122,183,40
140 DATA228,253,119,18,58,42,122,253,119,17,205,53,64,183,194
150 DATA14,64,253,110,52,253,102,53,253,126,18,61,203,39,95,22
160 DATA1,21,253,126,17,254,8,63,237,90,230,7,60,71,78,203,1
170 DATA203,9,16,252,203,193,71,203,9,203,1,16,252,113,253,110
180 DATA49,253,102,50,1,126,1,5,9,126,183,40,10,253,119,18,35
190 DATA126,253,119,17,24,179,205,23,64,183,194,14,64,24,133
200 DATA0,0,0
```

After you run program one , it will look this :-

```
10 REMPEEKLEN@2I39+8@INP
```

PROGRAM TWO , Leave line ten in memory and enter lines 20 to 280.

```
20 POKE30862,238:POKE30863,122:POKE30777,1
30 CLS:PRINT"    THIS SMALL PROGRAM PLACES"
40 PRINT"    LABELS INTO THE DIRECTORY"
50 PRINT"   FOR EASE OF IDENTIFICATION."
60 PRINT"    LABELS MAY BE UP TO 10"
70 PRINT"    CHARACTERS LONG."
80 PRINT@224,"ENTER LABEL      .:........":PRINT"ENTER TRACK      .."
90 PRINT"ENTER SECTOR    ..":PRINT"ENTER START ADDRESS    ...."
100 PRINT"ENTER END ADDRESS      ...."
110 A=PEEK(30744):IFA=1THENA=34ELSEA=98
120 POKE28910,A:PRINT@236,;:INPUTA$:IFA$="DIR......."THEN200
130 C=31273:PRINT@269,;:INPUTB$:GOSUB270
140 PRINT@301,;:INPUTB$:GOSUB270
150 PRINT@340,;:INPUTC$:GOSUB220:PRINT@372,;:INPUTC$:GOSUB220
160 A$=A$+"          "
170 X=USR(0):B=PEEK(31513)+256*PEEK(31514)-65536
180 FORI=0TO9:POKEB+I,ASC(MID$(A$,I+1,1)):NEXT:C=31273
190 FORI=10TO15:POKEB+I,PEEK(C):C=C+1:NEXT:X=USR(1):GOTO80
200 CLS:DIR:STATUS:PRINT"PRESS RETURN WHEN READY";
210 A$=INKEY$:IFINKEY$=CHR$(13)THEN30ELSE210
220 B$=RIGHT$(C$,2):GOSUB270:B$=LEFT$(C$,2):GOSUB270:RETURN
230 B1$=LEFT$(B$,1):GOSUB250:A=D*16:B1$=RIGHT$(B$,1):GOSUB250
240 A=A+D:POKEC,A:C=C+1:RETURN
250 D=ASC(B1$):IFD<58THEND=D-48ELSED=D-55
260 RETURN
270 IFB$=".."THENB$="00"
280 GOSUB230:RETURN
```

```
    ;BASIC CALL
    ;ORG.7AFFH
         PUSH HL
         DI
         CALL 4028H
         LD   C,32H
         LD   B,0FFH
         INC  B
         CALL 4038H
         IN   A,(13H)
         OR   A
         LD   A,4
         JP   M,400EH
         LD   A,(7921H)
         OR   A
         JR   NZ,A2
         CALL 402FH
         OR   A
         JP   NZ,400EH
         LD   (ADD),HL
    A1   CALL 400BH
         EI
         POP  HL
         RET
    ADD  DEFW 0FEC9H
    A2   CALL 4032H
         OR   A
         JP   NZ,400EH
         CALL 4011H
         OR   A
         JP   NZ,400EH
         LD   A,(31273)
         OR   A
         JR   Z,A1
         LD   (IY+12H),A
         LD   A,(31274)
         LD   (IY+11H),A
    A3   CALL 4035H
         OR   A
         JP   NZ,400EH
         LD   L,(IY+34H)
         LD   H,(IY+35H)
         LD   A,(IY+12H)
         DEC  A
         SLA  A
         LD   E,A
         LD   D,1
         DEC  D
         LD   A,(IY+11H)
         CP   8
         CCF
         ADC  HL,DE
         AND  7
         INC  A
         LD   B,A
         LD   C,(HL)
         RLC  C
    A4   RRC  C
         DJNZ A4
         SET  0,C
         LD   B,A
         RRC  C
    A5   RLC  C
         DJNZ A5
         LD   (HL),C
         LD   L,(IY+31H)
         LD   H,(IY+32H)
         LD   BC,017EH
         DEC  B
         ADD  HL,BC
         LD   A,(HL)
         OR   A
         JR   Z,A6
         LD   (IY+12H),A
         INC  HL
         LD   A,(HL)
         LD   (IY+11H),A
         JR   A3
    A6   CALL 4017H
         OR   A
         JP   NZ,400EH
         JR   A1
         NOP
         NOP
         NOP
```